**Amendments to the Claims:**

This listing of claims will replace all prior versions, and listings, of claims in the application:

**Listing of Claims:**

1. (Currently Amended) A compiling system embodied on a computer readable storage medium for compiling a markup language file into an executable application, the compiling system comprising:

a parser for parsing the markup language file and providing the compiling system with detailed token information;

a code generator for generating a language-independent tree of code expressions based on the token information, wherein the code expressions represent the markup file as a class; and

a compiler for compiling the code expressions to create the executable application.

2. (Original) The compiling system of claim 1, wherein the detailed token information comprises a tag.

3. (Original) The compiling system of claim 1, wherein the detailed token information comprises a property or event.

4. (Original) The compiling system of claim 1, wherein the detailed token information comprises a user code snippet.

5. (Original) The compiling system of claim 1, wherein the markup language file is associated with at least one code-behind file.

6. (Original) The compiling system of claim 5, wherein the compiler is configured to compile the markup language file and the code-behind file.

7. (Original) The compiling system of claim 1, wherein the executable application is an intermediate language application.

8. (Original) The compiling system of claim 1, further comprising a binary file generator for generating a binary file from non-code token information, wherein the binary file contains one record for each non-code token.

9. (Currently Amended) A compiling system embodied on a computer storage readable medium for compiling a markup language file into an executable application, the compiling system comprising:

a parser for parsing the markup language file and providing the compiling system with detailed token information including non-code token information to the compiling system, wherein the markup language file is associated with at least one code-behind file;

a binary file generator for generating a binary file from non-code token information, wherein the binary file contains one record for each non-code token; ~~and~~

a code generator for generating a language-independent code expression that represents the markup language file as a class; and

an application generator for compiling the code files into an executable application.

10. (Canceled)

11. (Currently Amended) The compiling system of claim 9[[ 10]], wherein the application generator combines the binary files into a single resource.

12. (Original) The compiling system of claim 9, wherein the detailed token information comprises a tag.

13. (Original) The compiling system of claim 9, wherein the detailed token information comprises a property or event.

14. (Original) The compiling system of claim 9, wherein the detailed token information comprises a user code snippet.

15. (Canceled)

16. (Currently Amended) The compiling system of claim 9[[ 15]], wherein the compiling system is configured to compile the markup language file and the code-behind file.

17. (Original) A method for compiling a markup language file into an executable application, the method comprising:

receiving a markup language file;

parsing the markup language file and providing a compiling system with detailed token information;

generating a language-independent tree of code expressions based on the token information, wherein the code expressions represent the markup language file as a class; and

compiling the code expressions to create the executable application.

18. (Original) The method of claim 17, further comprising receiving a code-behind file.

19. (Original) The method of claim 18, further comprising compiling the markup language file and the code-behind file.

20. (Original) The method of claim 17, further comprising providing a tag as detailed token information.

21. (Original) The method of claim 17, further comprising providing a property or event as the detailed token information.

22. (Original) The method of claim 17, further comprising providing a user code snippet as the detailed token information.

23. (Original) The method of claim 17, further comprising receiving a command to create an intermediate language application.

24. (Original) The method of claim 17, further comprising generating a binary file from non-code token information, wherein the binary file contains one record for each non-code token.

25. (Currently Amended) A computer readable storage medium storing the computer executable instructions for performing the method of claim 17.

26. (Currently Amended) A method for compiling a markup language file into an executable application, the method comprising:

parsing the markup language file and providing the compiling system with detailed token information including non-code token information;

generating a binary file from the non-code token information, wherein the binary file contains one record for each non-code token; ~~and~~

generating a language-independent code expression that represents the markup language file as a class; and

compiling the code expressions into an executable application.

27. (Canceled)

28. (Currently Amended) The method of claim 26[[ 27]], further comprising combining the binary files into a single resource.

29. (Currently Amended) The method of claim 26[[ 27]], further comprising providing a tag as the detailed token information.

30. (Currently Amended) The method of claim 26[[ 27]], further comprising providing a property or event as the detailed token information.

31. (Currently Amended) The method of claim 26[[ 27]], further comprising providing a user code snippet as the detailed token information.

32. (Currently Amended) The method of claim 26[[ 27]], further comprising receiving at least one code-behind file associated with the markup language file and compiling both the code-behind file and the markup language file.

33. (Currently Amended) A computer readable storage medium having computer executable instructions for performing the method of claim 26[[ 27]].